Insper

Ativos Digitais e Blockchain

Ricardo Rocha
Raul Ikeda

Objetivo

- Programando SmartContracts
- Interagindo com Contratos

Atenção: se você ainda não instalou os pacotes da aula passada, faça antes de começar essa aula

Cronograma

- Separação por mesas
- 1h - Implementação da atividade
- 15min – Discussão das perguntas
- 15min – Rotação e discussão
- 15min – Rotação e discussão
- 15min – Fechamento

My Own Bank[1]

Na aula de hoje vamos implementar um contrato que vai emular o básico de conta corrente de um banco.

O contrato vai realizar as seguintes operações:

- Depósito para conta interna
- Saque para conta externa
- Transferência entre contas internas

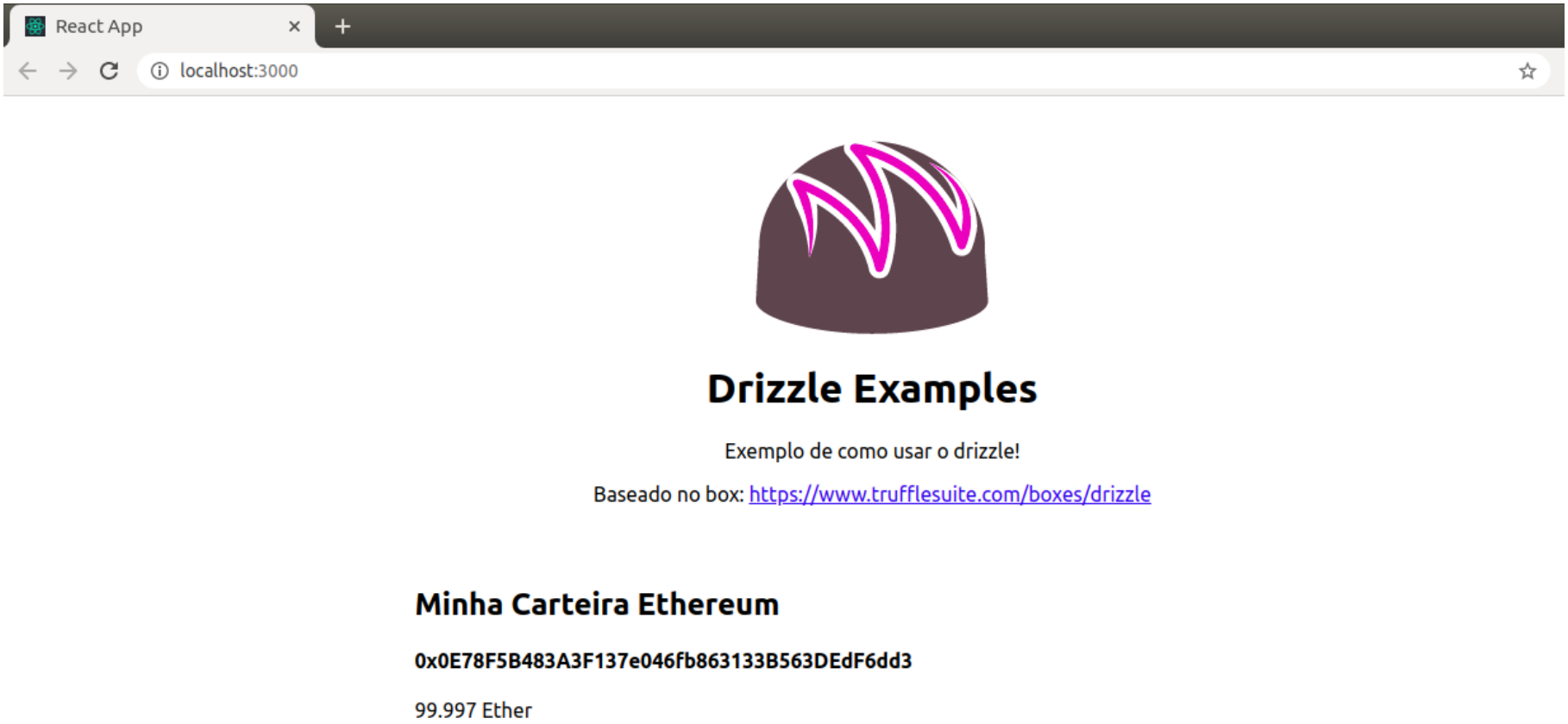
Pergunta do Milhão: Por que uma pessoa abriria uma conta no seu banco, sendo que ela já consegue realizar as operações acima?

[1] Baseado no Tutorial: https://www.trufflesuite.com/boxes/drizzle

Começando

1. Criar uma pasta
2. Abrir o Ganache e criar um Workspace na porta 7545
3. No prompt de comando apontando para essa nova pasta:
   - > truffle unbox raulikeda/AD-BOX
   - Café rápido, vai demorar um pouquinho
   - > truffle compile
   - > truffle migrate
   - > cd app
   - > node node_modules/react-scripts/scripts/start.js

Voilà


React App
localhost:3000
Drizzle Examples
Exemplo de como usar o drizzle!
Baseado no box: https://www.trufflesuite.com/boxes/drizzle
Minha Carteira Ethereum
0x0E78F5B483A3F137e046fb863133B563DEdF6dd3
99.997 Ether

Trabalhando no Contrato

- A tela contém informações da primeira conta do Ganache
- Não feche o prompt, deixe ele no canto, abra um novo prompt
- Agora vamos começar a rechear o contrato
- Abra o arquivo .\contracts\Bank.sol
  - De preferência com o Visual Studio ou outro editor do coração

Trabalhando no Contrato

- O contrato possui apenas um **atributo owner** e um **método constructor**
- Você deve implementar o seguinte atributo público:
  - **capital** do tipo **uint256**. Zerar a variável no construtor.

```
contract Bank {
    uint256 public capital;
    address public owner;
```

- Implementar também uma **struct Client**, com dois atributos:
  - **clientID** do tipo **address**
  - **balance** do tipo **uint256**

```
struct Client {
    address clientID;
    uint256 balance;
}
```

- Ainda, um “dicionário” **clients** que mapeia **address** em **Client**:

```
mapping(address => Client) public clients;
```

Foto do Momento

```
contract Bank {
    uint256 public capital;
    address public owner;

    struct Client {
        address clientID;
        uint256 balance;
    }

    mapping(address => Client) public clients;

    constructor() public payable {
        owner = msg.sender;
        capital = 0;
    }
```

Métodos – Todos ***public***

- deposit() - Deposita dinheiro na conta
  - Sem argumentos de entrada
  - payable
  - O método vai pegar o endereço do msg.sender e valor do msg.value e vai criar uma entrada em clients com o saldo inicial
  - Não precisa criar uma struct, pode preencher direto.
  - Deve ainda somar o valor à variável capital
- withdraw() - Saca dinheiro na conta
  - Argumento: amount do tipo uint256
  - Retorno: uint256
  - O método vai realizar um saque na conta do msg.sender se houver fundos. Nesse caso retorne o valor do saque, senão retorne 0
  - Use o transfer() e não use o *require*
  - Subtraia o amount do atributo capital

Métodos – Todos ***public***

- transfer() - Transfere dinheiro internamente entre contas
  - Argumentos: **beneficiary** do tipo **address** e **amount** do tipo **uint256**
  - Retorno: **true** se deu certo ou **false** caso contrário
  - A função irá transferir o valor do amount da conta do msg.sender para o beneficiary. Tudo deve ser interno e não pode haver transações fora na Blockchain.
  - Por ser interna, não modifica o atributo capital.

App

- No novo terminal, na pasta do projeto:
  - > truffle compile --all
  - > truffle migrate --reset
- Agora edite o arquivo: .\app\src\MyComponent.js
- Remova os comentários:
- Linha 44: **{/***
- Linha 89: ***/}**
- Salve o arquivo
- Olhe o browser novamente!

Voilà 2 – A Missão

Drizzle Examples

Exemplo de como usar o drizzle!

Baseado no box: https://www.trufflesuite.com/boxes/drizzle

Minha Carteira Ethereum

0x0E78F5B483A3F137e046fb863133B563DEdF6dd3

99.947 Ether

Saldo da Conta no Banco

Exemplo de como pegar um atributo do contrato com a minha própria chave.

Saldo Atual:

- **clientID**
  0x0000000000000000000000000000000000000000
- **balance**
  0

Realizando Operações

Abaixo algumas das interações possíveis com o contrato, baseado nos métodos disponíveis.

Deposito (1000 wei)

Submit

Saque

Quantidade Desejada Submit

Transferência

Conta Destino Quantidade Desejada Submit

Insper

App

- Agora teste livremente
- Acompanhe também os blocos no Ganache
- Modifique o endereço no browser para:
  - http://localhost:3000/?acc=1

Gabarito

```
contract Bank {
    uint256 public capital;
    address public owner;

    struct Client {
        address clientID;
        uint256 balance;
    }

    mapping(address => Client) public clients;

    constructor() public payable {
        owner = msg.sender;
        capital = 0;
    }

    function deposit() public payable {
        clients[msg.sender].clientID = msg.sender;
        clients[msg.sender].balance += msg.value;
        capital += msg.value;
    }
```

Gabarito

```
    function withdraw(uint256 amount) public returns (uint256) {
        if (clients[msg.sender].balance >= amount) {
            clients[msg.sender].balance -= amount;
            capital -= amount;
            msg.sender.transfer(amount);
            return amount;
        }
        return 0;
    }

    function transfer(address beneficiary, uint32 amount)
        public
        returns (bool)
    {
        if (clients[msg.sender].balance >= amount) {
            clients[msg.sender].balance -= amount;

            clients[beneficiary].clientID = beneficiary;
            clients[beneficiary].balance += amount;
            return true;
        }
        return false;
    }
}
```

Discussão em Mesas

- Como faríamos para implementar novas funcionalidade ao banco?
- Por exemplo:
  - Empréstimos
  - Financiamentos
  - Cartão de Crédito
  - Seguros
  - Etc
- Novamente: por que alguém utilizaria esse serviço? É confiável?
- Sobre as ideias de negócios apresentadas na aula retrasada, qual te motiva mais? Cada pessoa expõe seu ponto de vista sobre o assunto.

Projeto

1. Em grupos de 3 ou 4, vocês devem formatar um modelo de negócios usando o Blockchain. No máximo 2 pessoas de um mesmo curso.
2. Não pode ser apenas uma cryptomoeda. É preciso ter alguma atividade alvo atrelada ao projeto.
3. Você tem que apresentar a ideia do projeto e implementar usando uma linguagem de programação. Pode-se usar um framework ou infraestrutura em Cloud.
4. Primeiro Deadline: Proposta detalhada do projeto via README do Git até 19/Mai.
5. Realização de uma apresentação na data da Prova Final.

Próxima Aula

Microestrutura de Mercado